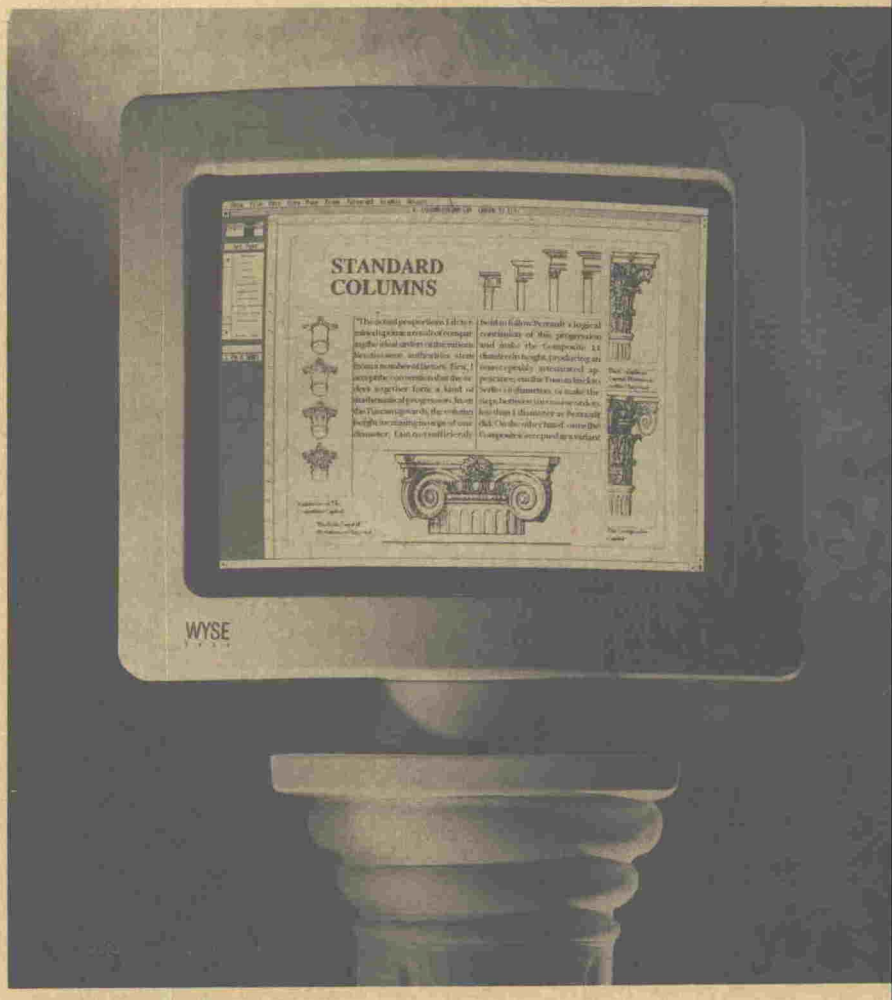
SOFT MAIL

COLUMN ERECTOR
จัดเอกสารให้เป็นคอลัมน์

หนึ่งในรูทีนสำคัญของโปรแกรมทางด้านเดสค์ท้อปพับลิชชิงและเวิร์ดโปรเซสเซอร์รุ่นใหม่



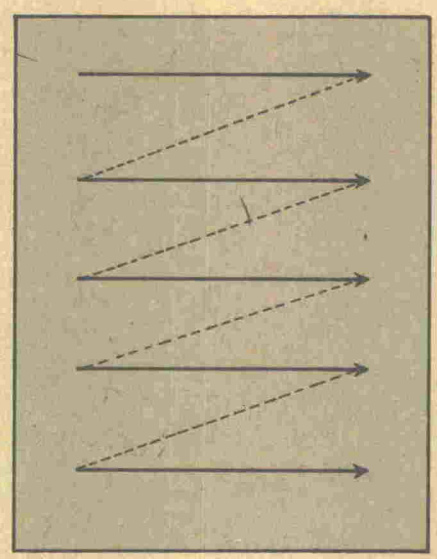
ถ้ามองย้อนกลับไปในอดีตจะพบว่า โปรแกรมประเภทเวิร์ดโปรเซสเซอร์ซึ่งช่วยงานในด้านการจัดเตรียมและพิมพ์เอกสารนั้น มีความสามารถกำหนดรูปแบบของเอกสารเพียงแค่ ตั้งกั้นหน้า กั้นหลัง ขอบบนและขอบล่างเท่านั้น แต่ก็ยังนับได้ว่าช่วยให้งานพิมพ์สะดวกขึ้นมากกว่าการใช้เครื่องพิมพ์ดีด ต่อมาจนถึงยุคที่มีโปรแกรมพวกเดสค์ท้อปพับลิชชิงที่สามารถรวมภาพ ตัดต่อ และอื่น ๆ อีก เหมือนกับมีโรงพิมพ์อยู่บนโต๊ะทำงาน ก็ได้มีการพัฒนาเวิร์ดโปรเซสเซอร์ให้ดีขึ้นเรื่อย ๆ หรือเรียกว่าตามเดสค์ท้อปพับลิชชิงมาติด ๆ นอกจากนี้เวิร์ดโปรเซสเซอร์บางตัวยังนำคุณสมบัติและความสามารถบางอย่างของเดสค์ท้อปพับลิชชิงมาใช้ ซึ่งก็นับว่าเป็นคู่แข่งที่สูสีกันเลยทีเดียว

ความสามารถอันหนึ่งของเดสค์ท้อปพับลิชชิงที่โปรแกรมประเภทนี้ไม่ว่าจะถูกหรือแพงจะต้องมีคือ การจัดข้อความให้อยู่ในรูปของคอลัมน์ตามจำนวนที่ต้องการ ซึ่งเวิร์ดโปรเซสเซอร์รุ่นใหม่หลายตัวจะมีคุณสมบัติข้อนี้ ดังนั้นสำหรับบทความนี้จะเสนอโปรแกรมทำหน้าที่จัดเอกสารให้เป็นคอลัมน์ ซึ่งในการใช้งานอาจจะพบว่าไม่ค่อยสะดวกนัก แต่จุดประสงค์หลักก็คือ เสนอหลักการทำงานของรูทีนประเภทนี้ ซึ่งพบได้ทั่วไปในโปรแกรมจัดการเกี่ยวกับเอกสาร และสามารถนำไปประยุกต์ใช้ในโปรแกรมที่ผู้อ่านได้พัฒนาขึ้นมาเอง

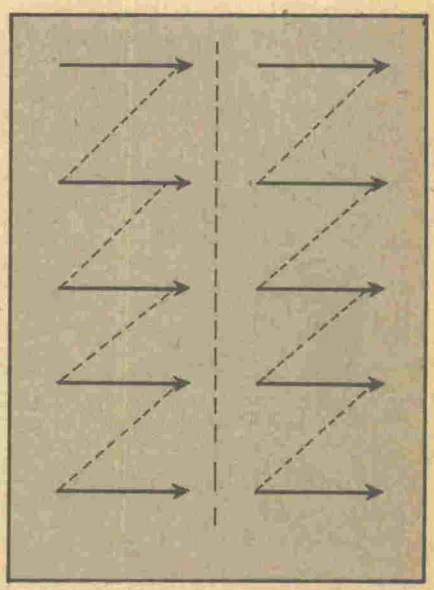
จัดคอลัมน์อย่างไร

ถ้ามองถึงการทำงานของโปรแกรมแล้ว หลายคนอาจคิดว่าคงจะยุ่งยากและมีความซับซ้อน จริง ๆ แล้วส่วนของโปรแกรมที่ทำหน้าที่จัดคอลัมน์นี้ มีขนาดเล็ก แต่ก็ทำงานได้เกินขนาดของมันทีเดียว

หลักการในการทำงานของมันก็มีเพียงแค่อ่านตัวอักษรจากเอกสาร แล้วนำมาใส่ไว้ในตำแหน่งที่ถูกต้องเท่านั้น โดยการอ่านจะเป็นการอ่านอักษรทีละตัวจากเอกสาร (หรือในที่นี้คือเท็กซ์ไฟล์) ดังทิศทางที่แสดงในรูปที่ 1 ซึ่งก็เหมือนกับที่เราอ่านหนังสือทั่ว ๆ ไป คือ อ่านจากซ้ายไปขวาและจากบนลงล่าง ส่วนการนำอักษรที่อ่านมาใส่ในตำแหน่งที่เหมาะสม ก็จะเริ่มจากการพิจารณาว่าความกว้างของหน้ากระดาษ(เป็นจำนวนตัวอักษร) และจำนวนคอลัมน์ที่ต้องการเป็นเท่าไร เมื่อทราบองค์ประกอบ 2 ตัวนี้แล้ว เราก็จะทราบว่าในคอลัมน์หนึ่งๆ จะมีความกว้างกี่ตัวอักษร การดำเนินการหลังจากนี้ก็เพียงแต่อ่านตัวอักษรมาใส่และทำการนับไปเรื่อย ๆ เมื่อเกินความกว้างของคอลัมน์ก็จะขึ้นบรรทัดใหม่ จนกระทั่งเกินความยาวของหน้ากระดาษ(นับเป็นบรรทัด)ก็เป็นการขึ้นคอลัมน์ที่สอง และจะทำงานในลักษณะเช่นนี้จนกว่าจะครบจำนวนคอลัมน์ต่อ 1 หน้า หรือหมดตัวอักษรที่จะถูกอ่านเข้ามา ดังรูปที่ 2 จากนั้นก็พิมพ์ออกทางเอาต์พุตที่ต้องการ



รูปที่ 1 : แสดงทิศทางการอ่านตัวอักษรจากเอกสาร



รูปที่ 2 : แสดงทิศทางการเขียนตัวอักษรที่อ่านลงในตำแหน่งที่เหมาะสม ซึ่งในภาพกำหนดให้มี 2 คอลัมน์

อธิบายโปรแกรม

จากหลักการทำงานที่กล่าวไปแล้ว เราต้องทำการเตรียมเนื้อที่ ซึ่งเพียงพอสำหรับข้อความขนาดหนึ่งหน้ากระดาษไว้สำหรับนำตัวอักษรมาจัดเรียงใหม่ โดยใช้ชื่อตัวแปรว่า page เป็นตัวแปรอะเรย์ 3 มิติ เพื่อที่เราจะสามารถอ้างถึงตัวอักษรใด ๆ โดยบอกถึงคอลัมน์ บรรทัดและลำดับของตัวอักษรในบรรทัดของคอลัมน์นั้น ต่อไปจะขออธิบายโปรแกรมดังในรูปที่ 3 แยกเป็นฟังก์ชันดังนี้

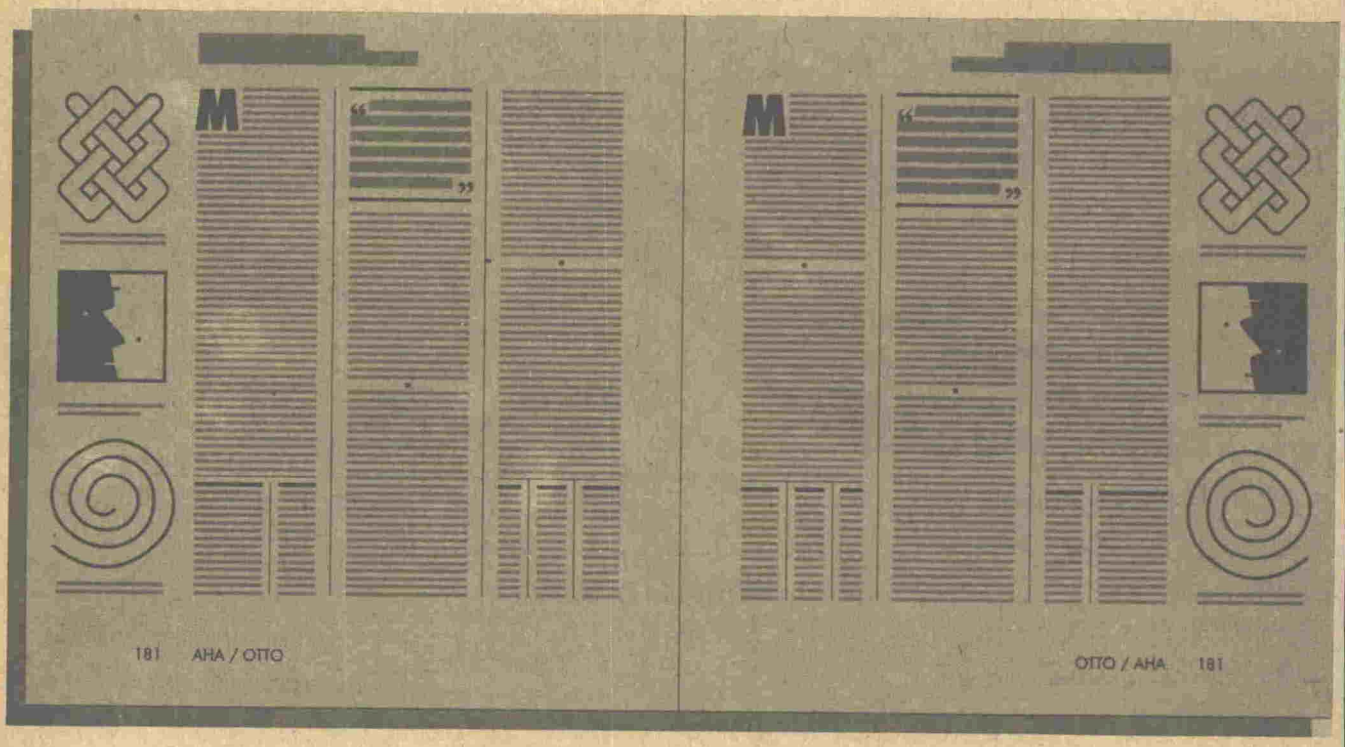
open_file ทำการเปิดไฟล์ทั้ง input และ output ตามอาร์กิวเมนต์ที่ป้อนเข้ามาในคอมมานด์ไลน์ซึ่งจะแจ้งให้ทราบถ้าไม่สามารถเปิดไฟล์ได้และจะหยุดการทำงาน

insert_space สำหรับเพิ่มช่องว่างให้เต็มบรรทัด เมื่อมีการขึ้นบรรทัดใหม่

arrange_page เป็นฟังก์ชันสำคัญ โดยมีหน้าที่นำตัวอักษรมาใส่ในตำแหน่งที่เหมาะสมในหน้ากระดาษ ซึ่งในโปรแกรมคือตัวแปรแบบอะเรย์ชื่อ page ตามหลักการที่ได้กล่าวมาแล้ว แต่จะมีการเปลี่ยน TAB ให้เป็นสเปซเพื่อให้ง่ายต่อการจัด และจะตรวจสอบว่าถ้าเป็น form-feed หรือ carriage return ก็จะขึ้นบรรทัดใหม่โดยเพิ่ม space จนครบบรรทัดด้วยฟังก์ชัน insert_space แต่ถ้าเป็นตัวอักษรอื่นจะเพิ่มตัวอักษรลงในตัวแปร page และเพิ่มค่าตัวแปร pos, เพิ่มค่า row เมื่อใส่จนครบบรรทัดและเพิ่มค่า col เมื่อครบบรรทัดในหนึ่งคอลัมน์ นอกจากนี้ในการขึ้นบรรทัดใหม่จะมีการเพิ่ม '–' ให้ถ้ายังไม่หมดคำ โดยตรวจสอบการหมดคำด้วยสเปซ และใช้ตัวแปร in_word ช่วยในการบอกสถานะ เมื่อใส่ตัวอักษรจนครบหนึ่งหน้าแล้ว ฟังก์ชันนี้จะรีเทอร์นค่า OK หรือ 1 เพื่อให้มีการเขียนข้อมูลใน page สู่ไฟล์ output แต่ถ้ายังไม่ครบจะรีเทอร์นค่า 0

write_page จะทำการเขียนตัวอักษรในตัวแปร page ไปยังไฟล์ output โดยเขียนแต่ละบรรทัดของทุกคอลัมน์เรียงกันไปจนครบ แล้วจึงขึ้นบรรทัดใหม่ หลังจากนั้นจะพิมพ์บรรทัดใหม่ของทุกคอลัมน์อีกจนครบหนึ่งหน้า จึงสั่งพิมพ์ form-feed เพื่อขึ้นหน้าใหม่ และกำหนดค่าให้กับตัวแปร pos, row และ col ในขณะที่เขียนตัวอักษรไปยังไฟล์ output จะทำการ initial ข้อมูลตัวนั้นด้วยโดยเปลี่ยนให้เป็น character 0

main เริ่มจากตรวจสอบจำนวนอาร์กิวเมนต์ในคอมมานด์ไลน์ ถ้าไม่ถูกต้องจะแสดงการใช้ให้ทราบและหยุดการ



ทำงาน ถ้าจำนวนอาร์กิวเมนต์ถูกต้องจะให้ตัวแปร col_num ซึ่งเก็บจำนวนคอลัมน์ต่อหน้ากระดาษเท่ากับอาร์กิวเมนต์ตัวที่ 4 ในคอมมานด์ไลน์แล้วตรวจดูว่าเกินจำนวนคอลัมน์ที่มากที่สุดที่กำหนดหรือไม่ จากนั้นจะเปิดไฟล์ input และ output และกำหนดให้ col, row และ pos เป็น 0 ทำการคำนวณความกว้างของคอลัมน์ให้กับตัวแปร col_width ต่อมาจะทำงานใน while-loop คืออ่านตัวอักษรมาจัด เมื่อครบหนึ่งหน้า(ฟังก์ชัน arrange_page return ค่า OK) จะเขียน page ลงในไฟล์ output ด้วยฟังก์ชัน write_page พร้อมกับเพิ่มค่าให้ตัวแปร page_num การทำงานจะทำไปในลักษณะนี้จนกว่าจะจบไฟล์ input เมื่อออกจากลูปแล้วจะตรวจว่ามีข้อมูลค้างอยู่ในตัวแปร page หรือไม่ ถ้ามีจะทำการเขียนข้อมูลที่ค้างนี้ลงในไฟล์ output จนหมดแล้วจึงทำการปิดไฟล์ input และ output

การใช้งาน

คราวนี้เราจะมาลองใช้งานกันเมื่อคอมพายล์และลิงค์จนได้เป็นไฟล์ .EXE แล้ว ก็สามารถรันบนดอสได้โดยพิมพ์

```
A>CE TEST.TXT OUT.TXT 3
```

หมายถึงไฟล์เอกสารคือ test.txt และไฟล์ output คือ out.txt โดยมีการทำเป็น 3 คอลัมน์ต่อหนึ่งหน้า เมื่อโปรแกรมทำงานเสร็จเราจะได้ไฟล์ out.txt นอกจากนี้เรายังสามารถให้โปรแกรมส่งผลการทำงานไปยังพรินเตอร์ได้โดยตรง โดยให้ไฟล์ output คือ PRN นั่นคือใช้

```
A>CE TEST.TXT PRN 3
```

แก้ไขตามต้องการ

ถ้าสังเกตตอนต้นของโปรแกรมจะเห็นว่ามีการ define สิ่งต่าง ๆ ไว้ที่ทำเช่นนี้ก็เพื่อความสะดวกในการนำไปดัดแปลงแก้ไขให้ตรงกับความต้องการของผู้ใช้ โดยอาจจะแยกส่วนนี้เป็นไฟล์ header (.h) ก็ได้ สิ่งที่กำหนดไว้ให้แก้ไขก็คือ

- **MAX_COL_PER_PAGE** คือจำนวนคอลัมน์ที่มากที่สุดในหนึ่งหน้า ในโปรแกรมกำหนดไว้เป็น 9 ซึ่งคิดว่าไม่น่าจะเกินนี้ แต่ถ้าใช้กระดาษขนาดใหญ่ขึ้น อาจต้องการจำนวนคอลัมน์เพิ่มขึ้น ก็สามารถแก้ไขค่านี้ได้ตามต้องการ
- **PAGE_LENGTH** เป็นจำนวนบรรทัดต่อหน้า
- **MAX_CHAR_PER_LINE** เป็นจำนวนตัวอักษรใน 1 บรรทัด ซึ่งก็สามารถแก้ไขให้เหมาะสมกับความกว้างของกระดาษได้เช่นกัน
- **V_LINE** เป็นเส้นแบ่งระหว่างคอลัมน์ ในโปรแกรมกำหนดไว้เป็น ':' ผู้ใช้อาจเปลี่ยนเป็น ' ' หรือตัวอักษรอื่น ๆ ได้ตามใจชอบ

สรุป

ดังจุดประสงค์ของโปรแกรมซึ่งได้กล่าวไปแล้ว หวังว่าบทความนี้คงจะเป็นประโยชน์ต่อผู้อ่าน ที่จะพัฒนาโปรแกรมจัดการเกี่ยวกับเอกสาร ซึ่งอาจต้องการรูทีนที่ทำหน้าที่จัดคอลัมน์ตามจำนวนที่ต้องการ เช่นอาจเป็นโปรแกรมสำหรับทำจดหมายข่าว (newsletter) หรืออื่น ๆ อย่างไรก็ตาม ประสิทธิภาพบางส่วนอาจถูกเพิ่มเข้าไปเพื่อให้โปรแกรมมีความสามารถเพิ่มขึ้น เช่นการพิมพ์เลขหน้าและการทำงานกับภาษาไทย โดยจัดการกับภาษาไทยแบบ 3 ระดับ รวมทั้งการตัดคำต่าง ๆ ด้วย แต่อย่างน้อยที่สุด บทความนี้คงจะทำให้ผู้อ่านได้รู้ถึงการทำงานของการนำเอกสารมาจัดเป็นคอลัมน์และไม่มองว่าเป็นสิ่งที่ยุ่งยากอีกต่อไป.

รูปที่ 3 : โปรแกรม CE.C

```
/****************************************/
/*            Column Erector            */
/*    Version 1.2     March 3, 1990     */
/****************************************/

#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>

#define MAX_COL_PER_PAGE    9
#define PAGE_LENGTH         60
#define MAX_CHAR_PER_LINE   80
#define V_LINE              '|'
#define BLANK               ' '
#define TAB                 '\t'
#define FORM_FEED           12
#define CR                  '\n'
#define OK                  1
#define YES                 1
#define NO                  0

FILE *document;
FILE *output;
char page[MAX_COL_PER_PAGE][PAGE_LENGTH][MAX_CHAR_PER_LINE];
char ch;
int  page_num=1;
int  col,row,pos;
int  col num;
int  col width;
int  in_word=NO;
int  buffer_empty=YES;

void open_file(char *argv[])
/**
 Open input and output files
**/
     { if ((document=fopen(argv[1],"rt")) == NULL)
          { printf("Can't open input-file:%s\n",argv[1]);
            exit(-1);
          }
       if ((output=fopen(argv[2],"wt")) == NULL)
          { printf("Can't open output-file:%s\n",argv[2]);
            exit(-1);
          }
     }

void insert_space()
/**
 Inserts spaces to end of line
**/
     { for (;pos < col width;pos++)
          page[col][row][pos]=BLANK;
     }

int arrange_page(char ch)
/**
 Put character to the appropriate position
**/
    { if ((ch == FORM_FEED) || (ch == CR))
         insert_space();
       if (ch == TAB)
         ch = BLANK;
       if (pos == (col_width-1))
         if (in_word)
            { page[col][row][pos++]='-';
              buffer_empty=NO;
            }
         else
            { insert_space();
              buffer_empty=NO;
            }
       if (pos < col_width)
          { page[col][row][pos++]=ch;
            buffer empty=YES;
```

รูปที่ 3 (ต่อ)

```
        }
      if (isspace(ch))
         in word=NO;
      else
         in word=YES;
       if (pos >= col width)
         { pos=0;
           row++;
         }
      if (row >= PAGE_LENGTH)
         { row=0;
           col++;
         }
      if (col >= col_num)
         return(OK);
      return(0);
    }

void write_page()
/**
 Writes arranged page to output file
**/
     { int line,column,position;

       for (line=0;line < PAGE_LENGTH;line++)
           { for (column=0;column < col_num;column++)
                 { for (position=0;position < col_width;position++)
                       { fprintf(output,"%c",page[column][line][position]);
                         page[column][line][position]=0;   /* initialize */
                       }
                   fprintf(output,"%c",V_LINE);
                 }
             fprintf(output,"\n");
           }
       fprintf(output,"%c\n",FORM_FEED);
       pos=row=col=0;
     }

main(int argc,char *argv[])
    { puts("Column Erector  Version 1.2");

      if (argc != 4)
         { puts("Usage: CE <in-file> <out-file> <number of column>");
           exit(-1);
         }

      col_num=atoi(argv[3]);

      if ((col_num > MAX_COL_PER_PAGE) || (col_num < 1))
         { puts("Invalid number of column!");
           exit(-1);
         }

      open_file(argv);
      col=row=pos=0;
      col_width=(MAX_CHAR_PER_LINE-col_num-1)/col_num;

      while (!feof(document))
            { if (buffer_empty)
                 ch=fgetc(document);
              if (arrange_page(ch) == OK)
                 { printf("Writing....page %-3d\n",page_num);
                   write_page();
                   page_num++;
                 }
            }
      if (page[0][0][0] != 0)
         { printf("Writing....page %-3d\n",page_num);
           write_page();
         }
      puts("Complete . . .");
      fclose(document);
      fclose(output);
    }
```